Journal of Theoretical and Applied Information Technology
15th January 2024. Vol.102. No 1
© 2023 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ALGORITHMIC COMPOSITION USING GATED RECURRENT UNIT FOR NATIONALISTIC MUSIC

KHAFIIZH HASTUTI [1], ERWIN YUDI HIDAYAT[2]

[1,2]Study Program in Informatics Engineering, Faculty of Computer Science, Universitas Dian Nuswantoro, Semarang, Indonesia

E-mail: [1]afis@dsn.dinus.ac.id, [2]erwin@dsn.dinus.ac.id

ABSTRACT

The study scrutinizes the implementation of the Gated Recurrent Unit within Recurrent Neural Networks for constructing nationalistic music. The GRU model demonstrates the capability to algorithmically emulate patriotic melodies from original compositions, thereby highlighting the transformational role of machine learning in crafting intricate musical structures. The effectiveness of the GRU model is further evaluated through the Turing Test, revealing a significant 46.5% misidentification rate. This evidence underlines the model's success in producing complex compositions that bear a striking resemblance to human-created pieces. Ultimately, these findings contribute to the broader understanding of GRU's potential in innovative music composition, thereby facilitating the enhancement of nationalism through the potent medium of music.

Keywords: *Algorithmic Composition, GRU, Nationalistic Music, Turing Test, LSTM*

1. INTRODUCTION

Nationalism, a distinct cultural affiliation, pervades various spheres including multinational firms, health behaviors, and public perceptions [1], [2], [3]. However, in many places like Indonesia, there is an alarming decline in nationalism, attributable to numerous factors. These include the commodification within consumer culture [4], negligence towards historical artifacts [5], insufficient preservation of cultural heritage [6], and the manifestation of conflicts and threats in individuals' lives [7]. Added to this, the reverberations of a global democratic regression impacting Indonesian democracy [8] and complex social and cultural dynamics intersecting with modernization and globalization processes [9] are compromising nationalism.

Amid these challenges, one potent medium to rekindle nationalism is music [10]. Music has demonstrated its capability to promote nationalism or patriotism, by imbuing cultural and national values and fostering national identity and pride [11], [12], [13]. Music can stimulate the human psyche as well as physical changes like pulse rate and breathing rhythm [14], [15]. Nationalistic music, crafted during periods of autonomy and self-rule, has been particularly influential in evoking patriotic sentiments and reminding the populace of their nation's history and struggles [16].

Traditional music composition, though, is an intricate, labor-intensive process [17]. Ideation and arrangement of elements often become a slog for composers. Therefore, a need for alternative and innovative approaches in music composition is evident, and algorithmic composition provides a potential solution [18], [19]. Utilizing patterns and principles in note sequences, it is possible to use computer algorithms combined with artificial intelligence and machine learning techniques to largely automate the composition process [20], [21], [22]. Examples of algorithm applications for automatic musical composition include rule-based for Gamelan music [23], Genetic Algorithms for creating melodic sequences capable of imitating human composition [24], and Neural Networks [25].

Advancements in machine learning, particularly the Gated Recurrent Unit (GRU) in Recurrent Neural Networks (RNNs), have shown promising outcomes in the realm of music composition [26], [27], [28], [29], [30]. For this research, we aim to use RNN with GRU variations to simulate human-like nationalist melodies. Specifically, we will explore patterns and temporal information found in data sequences of Indonesian nationalistic songs and strive to produce a melody indistinguishable from human compositions. The outcome of this research presents an opportunity to enhance the composition of nationalist music, contributing to efforts to bolster nationalism in contemporary society.

2. LITERATURE REVIEW

This section comprises the literature review related to the topic of the research.

2.1 Nationalistic Music

Music that is classified as nationalist serves as a genre that embodies the cultural, historical, and political identity of a nation or specific subgroup within a nation. This type of music often incorporates traditional musical structures, indigenous instruments, and lyrics that express nationalistic sentiments. Analyzing nationalist music requires examining its distinct characteristics and recurring elements while considering the cultural and historical context in which it arose. Furthermore, studying how music has been utilized to promote nationalistic ideology and foster unity provides valuable insights into the socio-political dynamics of society. In the Indonesian context, nationalistic music has played a substantial role in cultivating a shared identity among its diverse population. This is achieved through the incorporation of patriotic compositions such as *Indonesia Raya* and *Rayuan Pulau Kelapa* [31]

2.2 Algorithmic Composition

Algorithmic composition relies on computational methods to generate music varying from traditional composition that depends heavily on human intuition and creativity [32]. This technique boasts its origins to the mid-20th century with pioneers like Iannis Xenakis utilizing algorithms for crafting complex musical structures [33]. Later, advancements in technology equipped composers to delve into computer-programmed compositions, incorporating techniques such as rule-based grammar and cellular automata [34].

The advent of artificial intelligence and machine learning ignited significant transformations within algorithmic composition. Now, creations depend not solely on predetermined rules but can also learn from existing musical data [35]. This leads to more human-like compositions and has had profound impacts on the coupling of music and technology.

2.3 Recurrent Neural Network

The Recurrent Neural Network (RNN) is a Neural Network model distinguished by cyclic connections between units that allow an understanding of sequential data through the influence of past inputs on future data. Notably used in applications such as speech recognition [36], image recognition [37], language translation systems [38], text summarization [39], and even sequential data like music notation [40].

Simply put, the RNN architecture as depicted in Figure 1 receives an input x on the time dimension t i.e. x_t and hidden state h of the previous time dimension h_{t-1}. After the computing process is done, the system then generates an output value o and h on the t time as o_t and h_t. Hidden state in RNN models is used as internal memory, which is capable of temporarily storing information received by the system at the time t. Despite its storage potentials, RNN struggles with the Vanishing Gradient Problem, triggering swift changes in gradient values [41]. Consequently, alternatives like GRU or LSTM are often utilized due to constraints in practical RNN application.

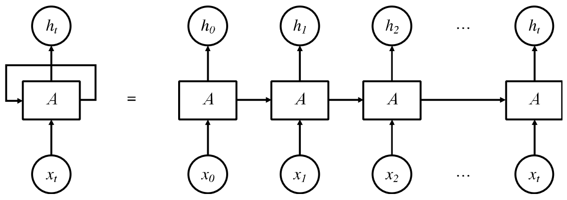

Figure 1: Architecture of the Recurrent Neural Network

2.4 Gated recurrent Unit

Gated Recurrent Units represent an adaptation of the conventional RNN model which facilitates the capturing of links across different time intervals [42]. The GRU architecture is less complex than LSTM, while still exhibiting similar functionalities. Differing from the tri-gate system of LSTM, GRUs solely integrate two gates and, as such, do not necessitate distinct memory cells for information preservation. Figure 2 shows the architecture of the GRU.

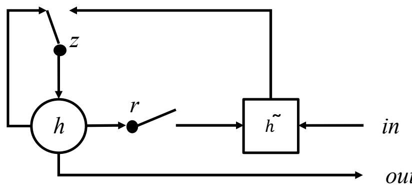

Figure 2: Architecture of the Gated Recurrent Units

Gated Recurrent Units computation primarily differs from pure RNN in the calculation of the hidden state (h_t). Prior to this, the GRU calculates three parameters, including two governing gates: the reset gate (r_t), which manages the blend of new inputs with previous information, and the update gate (z_t) orchestrating how previous information is processed further. In an ideal scenario, if the reset gate values were 1 and the update gate 0, GRU calculations would parallel pure RNN model output. The third parameter (g_t) represents a temporary hidden state ($\tilde{h}$) unaffected by previous calculations.

$$r_t=\sigma(U_r x_t + W_r h_{t-1}) \quad (1)$$
$$z_t=\sigma(U_z x_t + W_z h_{t-1}) \quad (2)$$
$$g_t=tanh\big(U_h x_t + W_h(r_t\, h_{t-1})\big) \quad (3)$$

Where, σ stands for the sigmoid function, x_t represents the input at time t, and h_{t-1} represents the hidden state at the previous time step. Each symbol W, U, r_t, and z_t are vectors and *tanh* stands for hyperbolic tangent function. After all parameters are calculated, the hidden state calculation process is performed using the following equation:

$$h_t=(1-z_t)h_{t-1}+z_t g_t \quad (4)$$

3. METHOD

This section describe the data collection and preprocessing, the proposed method, model training, music generation phase, and the model evaluation.

3.1 Data Collection and Preprocessing

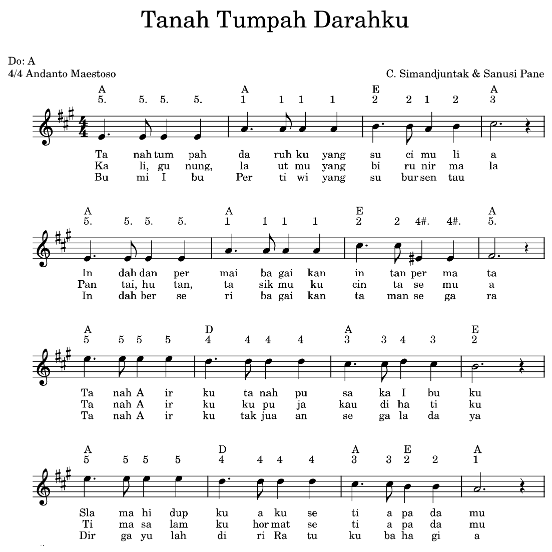

Figure 3. Tanah Tumpah Darahku song notation

Reference data for this study were extracted from songbook sources namely *Himpunan Lengkap Lagu-lagu Wajib Nasional dan Daerah* and *Kumpulan Terbaik & Terlengkap Lagu Wajib Nasional dan Daerah* comprising 45 national and regional songs, which included tracks like *Indonesia Pusaka* and *Bendera Merah Putih*. Preprocessing was indispensable for transposing numbered musical notations into a model-input-friendly format, which involved handling pitch and duration separately. An example of the preprocessing for *Tanah Tumpah Darahku* is illustrated in Figure 3.

The melody's pitch data handling involves recording each pitch value in a *pitch_train.txt* file based on numbered notation. High notes, low notes, and accidental notes are encoded using different symbols as (‘), (.), and (#) respectively. The symbol *0* denotes the absence of a note. Redundancy is avoided by only writing a note marked with a legato bow once, regardless of its repetition. Figure 4 displays illustrations of two identical tones that have been marked with the legato bow.

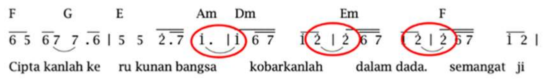

Figure 4. Two notes with legato bow

The outcomes of pitch depiction for the musical piece entitled *Tanah Tumpah Darahku* corresponding to every line of the aforementioned musical notation are ascertained as follows:

5. 5. 5. 5. 1 1 1 1 2 2 1 2 3 0
5. 5. 5. 5. 1 1 1 1 2 2 4#. 4#. 5. 0
5 5 5 5 4 4 4 4 3 3 4 3 2 0
5 5 5 5 4 4 4 4 3 3 2 2 1 0

Melody duration data is recorded in *duration_train.txt* file using decimal values representing the required beats or time for each note. For instance, a single beat is marked as 1, half a beat as 0.5, and two beats as 2. This setup allows for a precise analysis of the duration data for each notation line:

1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 3 1
1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 3 1
1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 3 1
1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 3 1

3.2 Proposed Method

This research employs a two-stage methodology. The training phase, as depicted in Figure 5, exploits 45 nationalistic songs stored in two files for pitch and duration data, with the aim to develop a model that learns from such music patterns. The composition stage, as depicted in Figure 6, uses short melodic sequences or random melody generation to create music, leading to a MIDI file with the new compositions. Preprocessing is required for transforming raw data into a suitable format for training. Each input consists of a sequence of *n* items with a target value, and both inputs and outputs should be vectors.

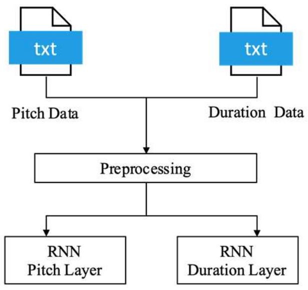

Figure 5: The Training Phase in the Proposed Method

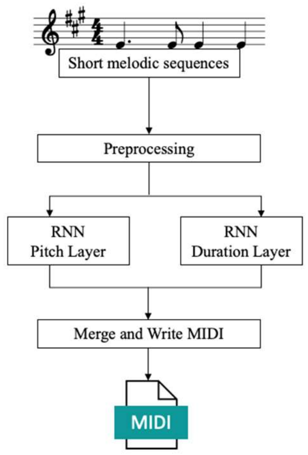

Figure 6: The Composition Phase in the Proposed Method

In preprocessing, full compositions are fragmented into smaller sequences with determined target values for training. If sequence length is set as *time_dim*, sequence data can be retrieved from the *i*-th to the (*i* + *time_dim*)-th data index. The target output is from the (*time_dim* + 1)-th data index. This process continues until all song data is used. An example is the first verse of W. R. Supratman's *Ibu Kita Kartini*:

1 2 3 4 5 3 1 6 1' 7 6 5

If the data is split into 4 tone sequences as the *time_dim* size and 1 tone sequence as the target, the input data obtained is presented in Table 1:

Table 1. The input data in four note sequence

i	Input Sequence	Target Output
0	1 2 3 4	5
1	2 3 4 5	3
2	3 4 5 3	1
3	4 5 3 1	6
4	5 3 1 6	1'
5	3 1 6 1'	7
6	1 6 1' 7	6
7	6 1' 7 6	5

In the preprocessing phase, data values in sequences are translated into a one-hot vector format, enhancing their representation. Each value becomes a 1 x *N* vector in the embedding model, where *N* signifies the vocabulary's size, holding all possible data value classes. Each vector element represents a distinct value from the vocabulary, excluding the actual value's index. For instance, in a seven-pitch musical scale per octave, the vocabulary comprises all conceivable note durations, totalling seven elements: vocabulary = {1, 2, 3, 4, 5, 6, 7}. When the tone value 2 is converted to the one-hot vector format, the resulting vector is:

$$\text{One\_hot vector}(2)=\begin{bmatrix}0\\1\\0\\0\\0\\0\\0\end{bmatrix}$$

Still in the preprocessing phase, the first element of the vector, a value of two, is disregarded. The second element is the actual value, while all other elements are set to zero. Examining a specific input data sequence (1,2,3,4,3,4,5,3) containing five unique values, we need an eight-dimensional array to hold it. Thus, the initial data is manipulated to form a matrix:

$$\begin{bmatrix}1&0&0&1&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&1&0&0&1\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&1&0\end{bmatrix}$$

Every input dataset, being a sequential arrangement of data, is processed so that each individual value is converted into an *N*-sized vector. This input data is represented by a 3D matrix of *I* x *S* x *N* dimensions, where *I* signifies the volume of training data, *S* indicates the count of sequences within an input, and *N* aptly reflects the vocabulary size.

3.3 Model Training

Once the preprocessing stage is completed, the resulting matrix will be used as the training dataset for the model. The procedural stages carried out by the model during the training process are described below.

1) Define model training parameters including epochs, hidden layers, and learning rate.
2) Initialize GRU variant RNN weights U (U_r, U_z, U_h) and W (W_r, W_z, W_h), V weight, and 4 bias weights (b_r, b_z, b_h, c).

3) Initialize a hidden state of 1 x *H* dimensions, where *H* equates to the size of the hidden layer.
4) Calculate values for GRU gates. Update gate (*z*) and reset gate (*r*) derive from sigmoid activation functions of specific sum values.
5) Ascertain temporary hidden state using tanh activation function.
6) Determine current time-step hidden state from update gate and temporary hidden state.
7) Sequentially repeat steps 4-6 for each input data sequence value.
8) Procure output vector containing probable output values using softmax activation function and Mean-Squared Error (MSE) for error quantification.

$$MSE=\frac{1}{2n}\ \Sigma_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \qquad (5)$$

where n denotes number of data points, Y_i is observed value, and $\hat{Y}_i$ is the predicted value.
9) Incrementally update all model weight values based on gradient and learning rate with Adam's optimizer.
10) Iterate steps 4-9 for all input data.
11) Replicate steps 4-10 for determined epochs.
12) Archive model weight values for future use in new music creation.

3.4 Music Composition Stage

Upon concluding the training phase, the model purportedly comprehends patterns and styles prevalent in Indonesian nationalistic music. The subsequent stage is music composition, where the generated melody undergoes a procedure akin to that of the training phase during the initial embedding process. The procedural operations for music composition are described below:

1) Initialize input parameters such as new note number and initial melody pieces.
2) Load model using precalculated weights from training phase.
3) Calculate GRU gate values using sigmoid functions on respective summations.
4) Acquire temporary hidden state using tanh activation function on resulting sum of input, reset gate, and bias.
5) Compute current hidden state using update gate and temporary hidden state, also accounting for previous timestep's hidden state.
6) Sequentially execute steps 3-5 for all data sequence values.
7) Generate output vector with softmax activation function on product of final hidden state and weight, summed with bias. Select highest output probability.
8) Implement steps 3-7 iteratively for preset number of new tones
9) Create new melody sequence post step 8 based on embedded training data patterns.

The pitch and duration components generated by the model are subsequently combined and integrated into a unified melodic structure, which is then encoded and saved as a MIDI file. For instance, the obtained outcomes of the pitches and durations are as follows:

Pitch = 1’ 7 6 5 6 5
Duration = 0.5 0.5 0.5 0.5 1 1

This suggests that the initial pitch data sequence's high *do* note (1') will be maintained for half a beat, its duration value denotes this. The final *la* note (6) of the last pitch data sequence will have a sound production duration of one tap, denoted by its respective duration value. Each value in the result sequence equates a pitch value and its time interval, especially when a pitch value is 0 that signals no note play in that specific interval. In the scope of MIDI production, sound signals are created using absolute pitch values. The *do-re-mi* tone requires conversion into an appropriate pitch value for MIDI file incorporation.

3.5 Model Evaluation

A Turing test will be employed to assess the model-generated music's quality without disclosing to the non-expert evaluators the origin of compositions. The evaluation set comprises a mixed collection of human-created and GRU-generated samples, presented in randomized order. Tester will assign a score of 0 or 1, representing the model or human composition, respectively. The test primarily aims to gauge the evaluators' ability to distinguish between human-composed and machine-produced music and collect subjective feedback on the evaluation difficulty.

4. RESULT AND DISCUSION

The parameters to perform model training are as follows: a) sequence size: 8; b) batch size: 24; c) number of epochs: 250; d) number of hidden dimension: 200. Table 2 below show the encoded pitch and duration data. Basically, the model training process in the GRU variant is to determine the 7 most appropriate weight values, namely U_r, U_z, U_h, W_r, W_z, W_h, and V and 4 biases b_r, b_z, b_h, and c. Firstly, the training phase is carried out on the RNN in the pitch layer, with pitch data as training data. The weights and biases of the pitch RNN are initialised randomly.

Table 2: Sample of Encoded Pitch and Duration Data

No.	Song Title	Pitch	Duration
1	Andika Bhayangkari	0 3 1 2 3 4 5 6 5 1 1' 1' 7 6 5 3 1 1 1' 1' 7 6 5 4 3 1 3 2 1 7. 1 0 1 4 3 2 3 4 5 4 3 1 6 5 4# 2 7 6 5 1 1' 1' 7 6 5 4 3 1 3 2 1 7. 1 3 3 2 1 7. 1	3 1 1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 2 1 0.5 0.5 1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 2 1 1 1.5 0.5 1 0.5 0.5 1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 3 1 1.5 0.5 1 1 4
2	Bangun Pemuda Pemudi	5. 3. 4. 5. 1 2 3 1 1 7. 2 1 7. 6. 5. 0 5. 3. 4. 5. 1 2 3 1 2 2 3 4# 5 0 2 2 2 3 3 4 3 4 3 3 2 1 3 2 0 5. 1 2 3 5 5 4 3 2 1 2 3 0 5. 1 2 3 5 5 4 3 2 3 2 1	1 0.75 0.25 1 0.75 0.25 2 2 1.5 0.5 0.5 0.5 0.5 0.5 3 1 1 0.75 0.25 1 0.75 0.25 2 2 1.5 0.5 1 1 3 1 1 0.75 0.25 1 1 1.5 0.5 2 1 0.75 0.25 1 1 3 1 1 0.75 0.25 1 1 2 1 0.75 0.25 2 2 3 1 1 0.75 0.25 1 1 2 1 0.75 0.25 2 2 4
3	Bendera Kita	3 2 1 2 3 1 5. 5 4 5 3 0 4 3 4 5 6 4 2 2 5 4# 6 5 0 3 2 1 2 3 1 5. 5 3 4 5 6 0 6 4 2 2 3 4 5 5 4 2 3 1 0	1 0.5 0.5 1 1 2 2 2 1 1 3 1 1.5 0.5 1 1 1.5 0.5 2 1 1 1 1 3 1 1 0.5 0.5 1 1 2 2 1.5 0.5 1 1 3 1 1.5 0.5 1 1 1.5 0.5 2 1.5 0.5 1 1 3 1

After all weight and bias values are randomly initialised, the training data is then fed into the model. Loss in the first iteration is calculated by MSE based on the difference between the resulting output and the target output to be achieved. By using equation (6), the MSE is obtained as follows:

$$\text{MSE} = \frac{\left(0\text{-}(\text{-}\ 0.0046)\right)^2 + \left(0\text{-}(\text{-}\ 0.0058)\right)^2 + \ldots}{2 \times 24}$$
$$\frac{+\left(0\text{-}(\text{-}\ 0.0053)\right)^2}{2 \times 24} = 0.45445$$

The value above is then used to calculate the gradient and change all the weight and bias values. Upon completing the training phase, which includes 250 epochs of weight and bias updates within each pitch and duration layer, the resultant values are conserved. These outcomes are later deployed during the music composition stage, indicating the allocation of weight and bias to the pitch layer. The following result shows the weights and biases of the GRU pitch layer:

U_r (8 x 200) =
$$\begin{bmatrix} -0.5168 & 1.3807 & \cdots & 0.0879 \\ -0.9756 & 0.1433 & \cdots & 1.1651 \\ 0.0113 & 0.6004 & \cdots & 0.2418 \\ 0.8141 & -0.4032 & \cdots & -0.2233 \\ -1.5003 & 1.2071 & \cdots & -1.0017 \\ 0.3677 & 0.8771 & \cdots & -0.9969 \\ -1.2141 & -0.4462 & \cdots & 0.4663 \\ 0.1929 & -0.8737 & \cdots & -0.0861 \end{bmatrix}$$

U_z (8 x 200) =
$$\begin{bmatrix} -0.0683 & -0.1761 & \cdots & 0.2227 \\ -0.2695 & 0.0440 & \cdots & -0.5168 \\ -0.2567 & 0.2873 & \cdots & -0.4508 \\ -0.4913 & -0.3130 & \cdots & 0.1909 \\ -0.9555 & 0.0962 & \cdots & -0.1060 \\ -0.8702 & 0.2500 & \cdots & 0.8441 \\ 0.1879 & -0.3179 & \cdots & -0.8508 \\ 0.3745 & 0.3205 & \cdots & -1.6519 \end{bmatrix}$$

U_h (8 x 200) =
$$\begin{bmatrix} -0.6427 & -0.0119 & \cdots & -0.2857 \\ 0.2032 & -0.1171 & \cdots & 0.1119 \\ -0.2546 & 0.2475 & \cdots & -0.1565 \\ -0.2569 & -0.2646 & \cdots & -0.6361 \\ 0.5948 & 0.6434 & \cdots & 0.3152 \\ 0.5359 & 0.3698 & \cdots & 0.0604 \\ 0.2219 & 0.5437 & \cdots & 0.2306 \\ -0.4765 & -0.1810 & \cdots & 0.2433 \end{bmatrix}$$

W_r (200 x 200) =
$$\begin{bmatrix} -0.1088 & -0.0883 & \cdots & 0.2094 \\ 0.0389 & -0.2485 & \cdots & -0.0647 \\ \vdots & \vdots & \ddots & \vdots \\ 0.1345 & -0.2250 & \cdots & -0.0195 \end{bmatrix}$$

W_z (200 x 200) =
$$\begin{bmatrix} -0.3532 & 0.2033 & \cdots & -0.3093 \\ 0.0931 & -0.1240 & \cdots & 0.1243 \\ \vdots & \vdots & \ddots & \vdots \\ 0.1695 & -0.3963 & \cdots & -0.2283 \end{bmatrix}$$

$$b_r(200) = \begin{bmatrix} -0.1656 \\ -0.1375 \\ -0.2551 \\ \vdots \\ 0.0886 \\ -0.1021 \end{bmatrix}; \; b_z(200) = \begin{bmatrix} -0.1307 \\ 0.1654 \\ -0.0694 \\ \vdots \\ 0.4118 \\ 0.0851 \end{bmatrix}$$

$$b_h(200) = \begin{bmatrix} -0.0610 \\ -0.0552 \\ 0.0358 \\ \vdots \\ 0.0780 \\ 0.2425 \end{bmatrix}; \; V\,(200 \text{ x } 1) = \begin{bmatrix} -0.0023 \\ -0.0056 \\ 0.0040 \\ \vdots \\ 0.1006 \\ 0.0890 \end{bmatrix}$$

c (1) = [0.0292]

In the testing phase, the desired sequence size determines the input values. For a sequence size of 8, as used in training, eight pitch and duration values are input to generate music. The utilized generation values are:

Pitch	= 5 1' 1' 7 6 5 1' 2'
Duration	= 1.5 0.5 1 1 1.5 0.5

$$h_t(1\ x\ 28\ x\ 200) = \begin{bmatrix} -0.0285 & -0.0298 & \cdots & 0.1240 \\ -0.0259 & 0.1032 & \cdots & 0.0494 \\ \vdots & \vdots & \ddots & \vdots \\ 0.1672 & 0.0519 & \cdots & 0.1009 \\ 0.1628 & 0.0568 & \cdots & 0.1006 \end{bmatrix}$$

$$\text{therefore } output = (h_t \times V) + c = \begin{bmatrix} 0.0438 \\ 0.1425 \\ -0.0100 \\ \vdots \\ 0.6598 \\ \vdots \end{bmatrix}$$

The produced output vector contains values for all possible outputs. The largest value and its corresponding index are identified, here being 0.6598 at index 14. According to the pitch vocabulary, index 14 signifies 3', hence the system has composed a new pitch of 3'. The same computation is executed on duration data, resulting in the highest output value of 0.4603 at index 2.

$$output = \begin{bmatrix} 0.0236 \\ 0.2169 \\ 0.4603 \\ \vdots \\ 0.0065 \\ \vdots \end{bmatrix}$$

In our duration vocabulary, index 2 equates to 1.5, implying a generated duration of 1.5 seconds per model prediction. For the next iteration, input consists of the recent eight sequence values. Utilizing a sequence length of nine from the initial iteration, the input for the following round comprises values ordered from two to nine. Here are the inputs for the model's second iteration:

Pitch	= 1' 1' 7 6 5 1' 2' 3’
Duration	= 0.5 1 1 1.5 0.5 1 1 1.5

The creation process is repeated whenever additional notes are required. Once nine new notes have been generated, the resulting melody is as follows:

Pitch	= 5 1' 1' 7 6 5 1' 2' 3’ 1’ 2’ 7 1’ 2’ 3’ 2’ 1’
Duration	= 1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 1.5

To listen to the final melody based on these pitch and duration values, the resulting data is converted into audio files in MIDI format. A Turing test, engaging ten individuals for evaluation, is then conducted to assess the model's efficacy in melody creation. Participants assess 20 MIDI audio files, consisting of 10 GRU-generated and 10 human-composed melodies, played in randomized order. Refer to Table 3 for sample music data used in the test, and Table 4 presents the Turing Test results.

Table 3: Sample of The Music Data for Testing

Code	Composer	Pitch Encoding	Pitch Duration	Musical Notation
L01	Machine	3 2 3 4 5 5 6 7 1’ 2’ 2’ 1’ 2’ 3’ 4’ 3’ 2’ 1’ 1’	2 1 1 2 2 2 1 1 2 2 2 1 1 2 2 2 1 1 2	3 . 2 3 | 4 . 5 . | 5 . 6 7 | 1> . 2> . | 2> . 1 2 | 3> . 4> . | 3> . 2 1 | 1 .
L03	Human: *Tanah Tumpah Darahku* (beats 1-8)	5. 5. 5. 5. 1 1 1 1 2 2 1 2 3 0 5. 5. 5. 5. 1 1 1 1 2 2 4#. 4#. 5. 0	1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 3 1 1.5 0.5 1 1 1.5 0.5 1 1 1.5 0.5 1 1 3 1	— — 5< .5< 5< 5 | 1 .1 1 1 | — 2 .2 1 2 | 3 . . 1 |
L12	Machine	5 5 3 4 5 6 5 3 1 3 2 1 3 4 2 4 2 3 4 2 4 2 5 6 5 4	1 1 1 0.5 0.5 4 0.5 0.5 0.5 0.5 1 1 4 1 1 1 1 1 1 4 1 1 1 1 2 2	5 5 3 $\overline{45}$ | 6 . . . | 53 $\overline{13}$ 2 1 | 3 . . . | 4 2 4 2 | 3 4 2 . | . . 4 2 | 5 6 5 . | 4 .
L13	Human: *Suburlah Tanah Airku* (beats 9-13)	1 1’ 1 6. 1 4 5 6 5 3 1 5. 1 2 3 2 3 6 5 3 2	1 1.5 0.5 0.5 0.5 0.5 0.5 1 0.5 0.5 0.5 0.5 0.5 0.5 1.5 0.5 0.5 0.5 0.5 0.5 2	_ -- -- 1 | 1> . 1 6<1 45 | 6 53 _ -- -- -- 15< 12 | 3 . 2 36 53 | 2 .

Table 4: Result of The Turing Test

MIDI Code	Tester										Total
	1	2	3	4	5	6	7	8	9	10	
L01	0	1	0	0	0	1	1	0	1	1	5
L02	1	0	1	0	0	0	1	1	0	0	4
L03	0	0	0	0	1	1	1	1	0	1	5
L04	1	1	0	1	0	1	0	0	1	1	6
L05	0	1	0	0	0	0	1	0	0	0	2
L06	0	0	0	1	0	0	0	0	0	1	2
L07	0	0	1	0	1	0	1	1	0	0	4
L08	1	0	1	0	0	1	1	0	1	1	6
L09	1	0	1	1	1	0	1	1	0	1	7
L10	1	1	0	1	1	1	1	0	1	1	8
L11	1	0	0	0	0	0	0	0	0	1	2
L12	1	0	0	1	1	1	1	0	1	0	6
L13	1	1	1	1	0	1	1	0	1	0	7
L14	0	0	1	0	1	0	0	1	1	1	5
L15	1	0	1	1	1	0	0	1	1	1	7
L16	1	0	1	1	0	1	0	1	1	1	7
L17	1	0	1	0	0	1	1	1	1	0	6
L18	0	1	0	1	0	1	1	0	1	1	6
L19	0	0	1	0	0	1	1	1	1	1	6
L20	1	1	0	1	0	0	1	1	0	1	6

$$\textit{Percentage of Failure} = \frac{\dfrac{\Sigma\ Total_i}{\textit{Number of Tester}}}{\textit{Number of Data}} \text{ x } 100\ \%$$

$$= \frac{\dfrac{12+7+10+10+7+11+14+10+12+14}{10}}{20} \text{ x } 100 = 46{,}5\%$$

Table 4 data suggests a high rate of incorrect predictions (46.5%) among the ten participants, indicating difficulty in discerning between GRU model-generated and human-composed melodies. All participants reported challenges in distinguishing between the two types of compositions.

This research certainly enhances our knowledge of algorithmic composition. It is also crucial to note that it comes with inherent constraints, such as the use of different algorithm and music dataset. This open future exploration and development in the study of aforementioned area.

REFERENCES:

[1] B. P. Dahal, "Nepalese nation, nationalism and identities in patriotic songs," Int. J. Learn. Dev., vol. 10, no. 4, p. 77, 2020, doi: 10.5296/ijld.v10i4.18135.

[2] J. Rius-Ulldemolins, "'The Great War' in the auto-making industry. Banal nationalism and symbolic domination and country-of-origin effect in consumer culture," J. Int. Consum. Mark., vol. 33, no. 1, pp. 98–112, Jan. 2021, doi: 10.1080/08961530.2020.1736706.

[3] H. Jia and X. Luo, "I wear a mask for my country: conspiracy theories, nationalism, and intention to adopt Covid-19 prevention behaviors at the later stage of pandemic control in china," Health Commun., vol. 38, no. 3, pp. 543–551, 2023, doi: 10.1080/10410236.2021.1958982.

[4] P. Bhagaskoro, R. U. Pasopati, and Syarifuddin, "Consumption and nationalism of Indonesia: between culture and economy," in The 3rd International Conference on Social and Political Sciences, 2017, vol. 129, pp. 211–213. doi: 10.2991/icsps-17.2018.45.

[5] H. J. R. Saragih, Suhirwan, A. Sarjito, Y. D. Damanik, and N. N. A. N. Avalokitesvari, "Management of defense heritage-based tourism to enhance youth nationalism and patriotism," J. Pertahanan, vol. 6, no. 2, pp. 278–285, 2020, doi: dx.doi.org/10.33172/jp.v6i2.847.

[6] V. R. Hadiz, "Democracy in indonesia: from stagnation to regression?," Bull. Indones. Econ. Stud., vol. 57, no. 1, pp. 135–137, 2021, doi: 10.1080/00074918.2021.1908877.

[7] M. Mietzner, "Sources of resistance to democratic decline: Indonesian civil society and Its trials," Democratization, vol. 28, no. 1, pp. 161–178, 2021, doi: 10.1080/13510347.2020.1796649.

[8] M. N. Iriansyah, W. S. Sumadinata, and Y. Djuyandi, "Pengaruh sikap nasionalisme pemuda terhadap keamanan di kota Bandung (studi pada siswa SMUN 3 dan SMUN 5 Bandung)," J. Anal. Sos. Polit., vol. 4, no. 2, pp. 86–95, 2020, doi: 10.23960/jasp.v4i2.59.

[9] B. H. Sukmadi, "The effect of leadership and ethics on motivation in the military," Int. Humanit. Appl. Sci. J., vol. 2, no. 3, pp. 49–58, Sep. 2019, doi: 10.22441/ihasj.2019.v2i3.05.

[10] C. Hill, "Poetic resistance: Karen long-distance nationalism, rap music, and YouTube," Int. J. Cult. Stud., vol. 25, no. 1, pp. 30–50, 2022, doi: 10.1177/13678779211027179.

[11] W. C. Ho, "Teachers' perspectives on cultural and national values in school music education between multiculturalism and nationalism in taiwan," Asia Pacific J. Educ., vol. 42, no. 4, pp. 627–640, 2022, doi: 10.1080/02188791.2021.1873101.

[12] T. Herbert, "Public military music and the promotion of patriotism in the British provinces, c. 1780-c. 1850," Ninet. Music Rev., vol. 17, no. 3, pp. 427–444, 2020, doi: 10.1017/S1479409819000594.

[13] C. Silver, "The sounds of nationalism: music, Moroccanism, and the making of Samy Elmaghribi," Int. J. Middle East Stud., vol. 52, no. 1, pp. 23–47, 2020, doi: doi.org/10.1017/S0020743819000941.

[14] O. Grewe, F. Nagel, R. Kopiez, and E. Altenmüller, How does music arouse "chills"? Investigating strong emotions, combining psychological, physiological, and psychoacoustical methods, vol. 1060. 2006. doi: 10.1196/annals.1360.041.

[15] H. Egermann and S. McAdams, "Empathy and emotional contagion as a link between recognized and felt emotions in music listening," Music Percept. An Interdiscip. J., vol. 31, no. 2, pp. 139–156, 2013, doi: 10.1525/mp.2013.31.2.139.

[16] A. Gilboa and E. Bodner, "What are your thoughts when the national anthem is playing? An empirical exploration," Psychol. Music, vol. 37, no. 4, pp. 459–484, May 2009, doi: 10.1177/0305735608097249.

[17] K. Hastuti, A. Azhari, A. Musdholifah, and R. Supanggah, "Building melodic feature knowledge of gamelan music using Apriori based on Functions in Sequence (AFiS) algorithm," Int. Rev. Comput. Softw., vol. 11, no. 12, pp. 1127–1137, 2016, doi: 10.15866/irecos.v11i12.10841.

[18] C. Ames, "The Markov process as a compositional model: a survey and tutorial," Leonardo, vol. 22, no. 2, pp. 175–187, 1989, doi: 10.2307/1575226.

[19] K. Jones, "Compositional applications of stochastic processes," Comput. Music J., vol. 5, no. 2, pp. 45–61, 1981, doi: 10.2307/3679879.

[20] David Cope, "Algorithmic music composition," in Patterns of Intuition, 1st ed., G. Nierhaus, Ed. Graz: Springer Dordrecht, 2015, pp. 405–416. doi: doi.org/10.1007/978-94-017-9561-6.

[21] H. B. Lopes, F. V. C. Martins, R. T. N. Cardoso, and V. F. dos Santos, "Combining rules and proportions: a multiobjective approach to algorithmic composition," in IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 2282–2289. doi: 10.1109/CEC.2017.7969581.

[22] Y. Feng and C. Le Zhou, "Advances in algorithmic composition," J. Softw., vol. 17, no. 2, pp. 209–215, 2006, doi: 10.1360/jos170209.

[23] K. Hastuti and K. Mustafa, "A method for automatic gamelan music composition," Int. J. Adv. Intell. Informatics, vol. 2, no. 1, pp. 26–37, 2016, doi: 10.26555/ijain.v2i1.57.

[24] C. K. Ting, C. L. Wu, and C. H. Liu, "A novel automatic composition system using evolutionary algorithm and phrase imitation," IEEE Syst. J., vol. 11, no. 3, pp. 1284–1295, 2017, doi: 10.1109/JSYST.2015.2482602.

[25] K. Hastuti, Azhari, A. Musdholifah, and R. Supanggah, "Rule-based and genetic algorithm for automatic gamelan music composition," Int. Rev. Model. Simulations, vol. 10, no. 3, pp. 202–212, 2017, doi: doi.org/10.15866/iremos.v10i3.11479.

[26] K. Choi, G. Fazekas, and M. Sandler, "Text-based LSTM networks for automatic music composition," in The First Conference on Computer Simulation of Musical Creativity, 2016, pp. 1–8. doi: doi.org/10.48550/arXiv.1604.05358.

[27] F. Colombo, S. P. Muscinelli, A. Seeholzer, J. Brea, and W. Gerstner, "Algorithmic composition of melodies with deep Recurrent Neural Networks," in 1st Conference on Computer Simulation of Musical Creativity, 2016, pp. 1–12. doi: 10.13140/RG.2.1.2436.5683.

[28] H. Chu, R. Urtasun, and S. Fidler, "Song From Pi: a musically plausible network for pop music generation," in The 5th International Conference on Learning Representations, 2016, pp. 1–9. doi: doi.org/10.48550/arXiv.1611.

[29] I.-T. Liu and B. Ramakrishnan, "Bach in 2014: music composition with Recurrent Neural Network," in International Conference on Learning Representation, 2014, pp. 1–9. doi: doi.org/10.48550/arXiv.1412.3191.

[30] Z. Sun et al., "Composing music with grammar argumented neural networks and note-level encoding," in Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, 2018, pp. 1864–1867. doi: doi.org/10.23919/APSIPA.2018.8659792.

[31] F. N. Rischa, S. Abdul, and Winarno, "The implantation of nationalism in globalization era using value clarification learning models," in The 1st International Conference on Education and Social Science Research, Jan. 2019, pp. 158–160. doi: 10.2991/icesre-18.2019.33.

[32] R. F. Cádiz, "Creating music with Fuzzy Logic," Frontiers in Artificial Intelligence, vol. 3. 2020. doi: doi.org/10.3389/frai.2020.00059.

[33] K. Kritsis, T. Kylafi, M. Kaliakatsos-Papakostas, A. Pikrakis, and V. Katsouros, "On the Adaptability of Recurrent Neural Networks for Real-Time Jazz Improvisation Accompaniment," Front. Artif. Intell., vol. 3, 2021, doi: 10.3389/frai.2020.508727.

[34] R. Rowe, "Representations, affordances, and interactive systems," Multimodal Technol. Interact., vol. 5, no. 5, pp. 1–8, 2021, doi: 10.3390/mti5050023.

[35] Y. Wang, "Music composition and emotion recognition using big data technology and Neural Network Algorithm," Comput. Intell. Neurosci., vol. 2021, p. 5398922, 2021, doi: 10.1155/2021/5398922.

[36] H. Dong, "Modeling and simulation of english speech rationality optimization recognition based on improved Particle Filter Algorithm," Complexity, vol. 2020, 2020, doi: 10.1155/2020/6053129.

[37] H. Wu et al., "Application of artificial intelligence in anatomical structure recognition of standard section of fetal heart," Comput. Math. Methods Med., vol. 2023, 2023, doi: 10.1155/2023/5650378.

[38] K. Balodis and D. Deksne, "FastText-Based intent detection for inflected languages," Information, vol. 10, no. 5. 2019. doi: 10.3390/info10050161.

[39] M. M. Rahman and F. H. Siddiqui, "An optimized abstractive text summarization model using peephole convolutional LSTM," Symmetry, vol. 11, no. 10. 2019. doi: 10.3390/sym11101290.

[40] D. Luitse and W. Denkena, “The great Transformer: examining the role of Large Language Models in the political economy of AI,” Big Data Soc., vol. 8, no. 2, Jul. 2021, doi: 10.1177/20539517211047734.

[41] W. Elmasry, A. Akbulut, and A. H. Zaim, “Deep learning approaches for predictive masquerade detection,” Secur. Commun. Networks, vol. 2018, 2018, doi: 10.1155/2018/9327215.

[42] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: encoder-decoder approaches,” in Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-8), Oct. 2014, pp. 103–111. doi: 10.3115/v1/W14-4012.